A. The claims are amended to now read as follows:

Claim 1. (Currently Amended) A method of operating a computer having a pipelined processor having a branch target buffer (BTB) comprising creating a recent entry queue, said recent entry queue comprising a set of branch target buffer (BTB) entries logically positioned in parallel with the branch target buffer (BTB), and organizing the recent entry queue as a FIFO queue, ~~comparing an entry to be written into the BTB against valid entries within the recent entry queue, blocking an entry matching an entry within the recent entry queue from being written into the BTB, searching the recent entry queue to detect looping branches, comparing the branch to determine if it was recently written into the queue, determining if the branch is backwards branching whereby a looping branch is detected, and if a looping branch is detected that is not predicted thereafter delaying a decode,and writing an entry into the BTB when it is also written into the recent entry queue~~ said Branch Target Buffer (BTB) and said recent entry queue being set associative and said recent entry queue being logically defined as a subset of the Branch Target Buffer (BTB) and coupled to track the last number of branches entered into said BTB and also the most recent entries thereby allowing a comparison of recent entries of said recent entry queue to said BTB and for blocking duplicate entries from being installed into the BTB and said recent entry queue by examining the contents of the recent entry queue for such duplicate entries prior to a write into said BTB and said recent entry queue and in addition for allowing a decode to be stalled by a defined amount of cycles such that a branch of interest can be delayed from decoding in order to allow a given entry in the BTB to be detected in time for future decodes of said branch of interest.

2.-3. (Canceled)

4. (Currently Amended) The method of claim ~~[[3]]~~ 1 wherein the recent entry queue is fully associative for reading.

5.-7. (Canceled)

8. (Originally Presented) The method of claim 1 comprising searching the BTB for a next predicted branch and evaluating the recent entry queue while the BTB is being indexed.

9. (Originally Presented) The method of claim 8 wherein the recent entry queue maintains a depth up to the associativity of the BTB whereby while the BTB is indexed, the recent entry queue positions are input to comparison logic.

10. (Previously Amended) The method of claim 8 comprising searching the recent entry queue for a matching branch in parallel to searching BTB output.

11. (Previously Amended) The method of claim 10 comprising creating hit detect logic to support the associativity of the BTB.

12. (Originally Presented) The method of claim 8 comprising using a subset of the recent entry queue as a subset of the BTB.

13. (Originally Presented) The method of claim 12 comprising fast indexing recently encountered branches.

14. (Previously Amended) The method of claim 12 comprising searching the complete recent entry queue to block duplicate BTB writes.

15. -18. (Canceled)

19. (Currently Amended) The method of claim ~~[[[18]]]~~ 1 comprising delaying decode until a fixed number of cycles.

20. (Currently Amended) The method of claim ~~[[[19]]]~~ 1 comprising delaying decode until the BTB predicts a branch.

21. (Originally Presented) The method of claim 1 comprising staging writes to the BTB in the recent entry queue.

22. (Previously Amended) The method of claim 21 comprising delaying a write and placing the write in the recent event queue.

23. (Previously Amended) The method of claim 22 comprising detecting a predicted branch while its BTB write is temporarily staged in the recent entry queue.

24. (Currently Amended) A computer having a pipelined processor comprising comparing a branch target buffer (BTB) with a recent entry queue, said recent entry queue comprising a set of branch target buffer (BTB) entries logically positioned in parallel with the branch target buffer (BTB) , said computer organizing the recent entry queue as a FIFO queue~~, comparing an entry to be written into the BTB against valid entries within the recent entry queue, blocking an entry matching an entry within the recent entry queue from being written into the BTB, searching the recent entry queue to detect looping branches, comparing the branch to determine if it was recently written into the queue, determining if the branch is backwards branching whereby a looping branch is detected, and if a looping~~

~~branch is detected that is not predicted thereafter delaying a decode, and writing an entry into the BTB when it is also written into the recent entry queue~~ said Branch Target Buffer (BTB) and said recent entry queue being set associative and said recent entry queue being logically defined as a subset of the Branch Target Buffer (BTB) and coupled to track the last number of branches entered into said BTB and also the most recent entries thereby allowing a comparison of recent entries of said recent entry queue to said BTB and for blocking duplicate entries from being installed into the BTB and said recent entry queue by examining the contents of the recent entry queue for such duplicate entries prior to a write into said BTB and said recent entry queue and in addition for allowing a decode to be stalled by a defined amount of cycles such that a branch of interest can be delayed from decoding in order to allow a given entry in the BTB to be detected in time for future decodes of said branch of interest.

25. (Originally Presented) The computer of claim 24 wherein the recent entry queue comprises a set of branch target buffer (BTB) entries.

26. (Canceled) ~~The computer of claim 25 wherein the recent entry queue is a FIFO queue.~~

27. (Previously Amended) The computer of claim 26 wherein the recent entry queue is fully associative for reading.

28. (Currently Amended) A program product comprising a computer readable medium having computer readable code thereon for controlling and configuring a computer having a pipelined processor and a branch target buffer (BTB) to create a recent entry queue, said recent entry queue comprising a set of branch target buffer (BTB) entries logically positioned in parallel with the branch target buffer (BTB)), organizing the recent

entry queue as a FIFO queue, ~~comparing an entry to be written into the BTB against valid entries within the recent entry queue, blocking an entry matching an entry within the recent entry queue from being written into the BTB, searching the recent entry queue to detect looping branches, comparing the branch to determine if it was recently written into the queue, determining if the branch is backwards branching whereby a looping branch is detected, and if a looping branch is detected that is not predicted thereafter delaying a decode, and writing an entry into the BTB when it is also written into the recent entry queue.~~
said Branch Target Buffer (BTB) and said recent entry queue being set associative and said recent entry queue being logically defined as a subset of the Branch Target Buffer (BTB) and coupled to track the last number of branches entered into said BTB and also the most recent entries thereby allowing a comparison of recent entries of said recent entry queue to said BTB and for blocking duplicate entries from being installed into the BTB and said recent entry queue by examining the contents of the recent entry queue for such duplicate entries prior to a write into said BTB and said recent entry queue and in addition for allowing a decode to be stalled by a defined amount of cycles such that a branch of interest can be delayed from decoding in order to allow a given entry in the BTB to be detected in time for future decodes of said branch of interest.

29. (Canceled)

30. (Currently Amended) The program product of claim ~~[[[29]]]~~ 28 further comprising code for organizing the recent entry queue as a FIFO queue.

31. (Previously Amended) The program product of claim 28 comprising code for making the recent entry queue fully associative for reading.

32. – 33. (Canceled)

34. (Currently Amended) The program product of claim ~~[[[32]]]~~ 28 further comprising code for writing an entry into the recent entry queue when the entry is written into the BTB.

35.-37. (Canceled)

38. (Previously Amended) The program product of claim ~~[[[37]]~~ 28 comprising code for creating hit detect logic to support the associativity of the BTB.

39. (Previously Amended) The program product of claim ~~[[[35]]]~~ 28 comprising code for using a subset of the recent entry queue as a subset of the BTB.

40. (Originally Presented) The program product of claim 39 comprising code for fast indexing recently encountered branches.

41. (Previously Amended) The program product of claim 39 comprising code for searching the complete recent entry queue to block duplicate BTB writes.

42.- 45. (Canceled)

46. (Currently Amended) The program product of claim ~~[[[45]]] 28~~ 28 comprising code for delaying decode until a fixed number of cycles.

47. (Originally Presented) The program product of claim 46 comprising code for delaying decode until the BTB predicts a branch.

48. (Originally Presented) The program product of claim 28 comprising code for staging writes to the BTB in the recent entry queue.

49. (Previously Amended) The program product of claim 48 comprising code for delaying a write and placing the write in the recent event queue.

50. (Previously Amended) The program product of claim 49 comprising code for detecting a predicted branch while its BTB write is temporarily staged in the recent entry queue.